REMARKS

Reconsideration of the application is respectfully requested.

Amendments to 2 paragraphs in the specification requested in the last reply was not entered because they referred to incorrect page and line numbers. In this reply, corrected page and line numbers for these paragraphs are submitted.

The Office Action provides that the specification includes derogatory remarks. Applicant was not aware of and certainly did not intend on making any disparaging remarks about the current state of the art. Applicant, however, has no objections in amending the paragraphs in the background section as suggested in the Office Action. Accordingly, the paragraphs on page 1 and 2 are amended.

With regard to the objection of claims 12 and 14 as being duplicates, claims 12 and 14 claim a syntax that has the positions of the identifiers reversed. That is, claim 12 recites a syntax that has the object type appearing first in the syntax. Claim 14 recites a syntax that has the object identifier appearing first in the syntax. Accordinlgy, Applicant believes that claims 12 and 14 are not duplicates.

With regard to the 35 U.S.C. §112 rejection of claims 20 and 21, these claims are amended to recite "compiler or interpreter."

With regard to the rejection of claims 1, 7, 9, 17 under 35 U.S.C. §102(b) as allegedly being anticipated by Berners-Lee, it is submitted that Bernes-Lee does not disclose or suggest every element claimed in independent claims 1 and 17 as amended. For instance, Bernes-Lee does not disclose or suggest a *programming language syntax*. Bernes-Lee does not disclose or suggest to use its protocol in a programming language "throughout a life of a program as a syntax for referencing an object in the program." For example, Bernes-Lee does not disclose or suggest to use its protocol expression as an expression for identifiers. Identifiers in a program usually can be assigned values. Bernes-Lee's protocol, however, cannot be assigned a value. Accordingly, it is submitted that Berners-Lee does not anticipate claims 1, 7, 9, and 17.

With regard to the rejection of claims 1, 2, 11-16, 18, 20, and 21 under 35 U.S.C. §102(b) as allegedly being anticipated by Staugaard, and the rejection of claims 1, 11-16, and 18-21 under 35 U.S.C. §102(b) as allegedly being anticipated by JLE, it is submitted that neither Staugaard nor JLE discloses or suggests every element claimed. For example, in Staugaard and JLE's syntax, explicitly type declaration cannot be used more than once in a program.

Accordingly, Staugaard and JLE doe not disclose or suggest every element claimed in 1, 2, 11-16, and 18-21.

With regard to the rejection of claims 3-6 and 8 under 35 U.S.C. §103(a) as allegedly being unpatentable over Staugaard as applied to claim 1, and further in view of "readme", it is submitted that Staugaard and readme fail to disclose, suggest, or teach at least that the object identifier name having the object type indicator is used throughout a life of a program as a syntax for referencing an object in the program. Accordingly, it is submitted that these claims are not obvious over Staugaard and readme.

With regard to the rejection of claim 10 under 35 U.S.C. §103(a) as allegedly being unpatentable over Berners-Lee, it is submitted that Berners-Lee does not disclose, suggest, or teach claim 10. For example, there is no motivation or desirability suggested in Berners-Lee to use its URL as a syntax for referencing an object in a program. Accordingly, it is submitted that these claims are not obvious over Berners-Lee.

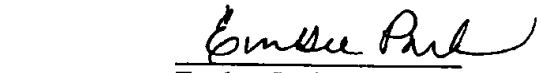
On May 29, 2003, a telephone interview was conducted between Examiner Eric Kiss of the USPTO and Eunhee Park, applicant's representative. Although no agreement was reached, Applicant appreciates Examiner's courtesy of granting that interview. During the interview, the Examiner explained that Fortran 77 allows implicit declaration of variables that can be used throughout a program. Fortran 77, for example, allows any variables beginning with letters i, j, or k to have default integer types, if they are not expressly declared as otherwise. It is submitted, however, that Fortran 77's implicit declaration scheme does not disclose, suggest, or teach the object type declaration syntax claimed in amended independent claims 1, 16, 17, and in new claim 22. For example, Fortran 77 does not disclose, suggest, or teach to include "the joint attribute used by an interpreter or a compiler of the programming language to separate the type declaration from the object identifier to determine an object type of an object being declared in the object identifier when the object identifier having the type declaration is read, the determining done without requiring additional referencing each time the object identifier having the type declaration is read" and allow "object identifier with the embedded type declaration to be used throughout a life of a program as a syntax for referencing an object in the program," as claimed in claim 1. Further, Fortran 77's implicit declaration scheme does not disclose, suggest, or teach to "embedding an object type indicator with an object identifier name, the object type indicator and the object identifier name delineated by a predefined symbol, wherein a machine uses the predefined symbol to separate the object type indicator and the object identifier name to

identify an object type for the object identifier name and the object identifier name having the object type indicator is used throughout a life of a program as a syntax for referencing an object in the program," as claimed in claim 16. Similarly, Fortran 77 does not disclose, suggest, or teach "prepending an object type followed by a predefined symbol to an object identifier string, the object type, the predefined symbol, and the object identifier string forming a symbol name to be carried throughout a life of a program as a syntax for referencing an object in the program, wherein a machine interpreting the symbol name in the program uses the predefined symbol to delineate the object type from the object to determine the object type," as claimed in claim 17.

Rather, Fortran 77's implicit declaration scheme appears to allow variable names begining with certain characters to have a default type. It is no consequence to Fortran 77 compiler what other characters follow a first letter in a variable. In Fortran 77, so long as a variable name begins with certain character, they are implicitly typed as a predetermined type; that is, if a variable name begins with i, j, or k, they are typed as integers. In contrast, the claims in the present application are directed to explicit object type declaration syntax where the type, any type, is embedded in the object name and delineated by a predefined symbol.

For at least the foregoing reasons, it is submitted that Fortran 77 or any other references cited in the office actions do not disclose, suggest, or teach the claims in the present application. This communication is believed to be fully responsive to the Office Action and every effort has been made to place the application in condition for allowance. If a telephone interview would be of assistance in advancing prosecution of the subject application, Examiner is requested to telephone the number provided below.

Respectfully submitted,



Eunhee Park
Registration No. 42,976
Baker & McKenzie
805 Third Avenue
New York, NY 10022
Telephone (212) 751-5700
Facsimile (212) 759-9133